(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第6670312号
(P6670312)
(45)発行日　令和2年3月18日(2020.3.18)　(24)登録日　令和2年3月3日(2020.3.3)

(51)Int.Cl.　F I
G06F 9/54 (2006.01)　G06F 9/54 F
G06F 9/54 D

請求項の数 33　(全 20 頁)

(21)出願番号　特願2017-534604(P2017-534604)
(86)(22)出願日　平成27年12月17日(2015.12.17)
(65)公表番号　特表2018-500688(P2018-500688A)
(43)公表日　平成30年1月11日(2018.1.11)
(86)国際出願番号　PCT/US2015/066303
(87)国際公開番号　WO2016/106064
(87)国際公開日　平成28年6月30日(2016.6.30)
審査請求日　平成30年7月6日(2018.7.6)
(31)優先権主張番号　14/581,417
(32)優先日　平成26年12月23日(2014.12.23)
(33)優先権主張国・地域又は機関
米国(US)

(73)特許権者　517219166
ドキュメント　ストレージ　システムズ,　インコーポレイテッド
アメリカ合衆国　フロリダ　３３４０８,　ジュノ　ビーチ,　ユーエス　ハイウェイ　１　１２５７５,　スイート　２００
(74)代理人　100078282
弁理士　山本　秀策
(74)代理人　100113413
弁理士　森下　夏樹
(74)代理人　100181674
弁理士　飯田　貴敏
(74)代理人　100181641
弁理士　石川　大輔

最終頁に続く

(54)【発明の名称】動的サービス展開のためのコンピュータ可読格納媒体およびそれを利用するための方法およびシステム

(57)【特許請求の範囲】
【請求項１】
メモリに結合される、少なくとも１つの処理ユニットを備える、コンピュータハードウェアシステムであって、前記メモリは、コンピュータ実行可能命令で符号化されており、前記コンピュータ実行可能命令は、実行されると、前記少なくとも１つの処理ユニットに、
公開エンドポイントにおいて、クライアントメッセージを受信することであって、前記メッセージは、第１のアプリケーションプログラミングインターフェースの第１の記述に従ってフォーマットされており、前記メッセージは、動作を規定する命令と、コンピュータハードウェアシステムが前記動作を実行することを可能にするパッケージを示す情報とを含む、ことと、
第２の記述に従って前記メッセージからの前記命令を解析することと、
前記第１のアプリケーションプログラミングインターフェースの機能を介して、前記パッケージに前記命令をパスすることであって、前記パッケージは、第２のアプリケーションプログラミングインターフェースおよび論理を有し、前記パッケージは、複数のパッケージのうちの１つである、ことと、
前記第２のアプリケーションプログラミングインターフェースを経由して、前記動作および前記論理に基づく結果を受信することと、
前記結果に基づいて前記クライアントメッセージに応答することと
を行わせ、

前記第１の記述は、多次元配列を備える、システム。
【請求項２】
前記第１のアプリケーションプログラミングインターフェースによって提供される抽象化の程度は、前記第２のアプリケーションプログラミングインターフェースによって提供される抽象化の程度よりも広い、請求項１に記載のシステム。
【請求項３】
前記多次元配列は、ジャグ文字列配列である、請求項１に記載のシステム。
【請求項４】
前記第１および第２の記述は、インターフェース記述言語から形成される、請求項１に記載のシステム。
【請求項５】
前記インターフェース記述言語は、ウェブサービス記述言語である、請求項４に記載のシステム。
【請求項６】
前記クライアントメッセージは、プレーンテキストでフォーマットされている、請求項１に記載のシステム。
【請求項７】
プロセッサと、
複数の公開エンドポイントと、
メモリと
を備えるコンピュータハードウェアシステムであって、前記メモリは、
パッケージアプリケーションプログラミングインターフェースを実装する複数のパッケージと、
サービスアプリケーションプログラミングインターフェースの機能性の記述と、
コンピュータ実行可能命令を備えるサービス展開エンジンであって、前記コンピュータ実行可能命令は、実行されると、前記プロセッサに、前記サービスアプリケーションプログラミングインターフェースの前記機能性の前記記述を修正することなく、前記複数のパッケージを複数のエンドポイントに展開することを行わせる、サービス展開エンジンと
を備え、前記サービスアプリケーションプログラミングインターフェースは、コンピュータ実行可能命令を備え、前記コンピュータ実行可能命令は、実行されると、前記複数の公開エンドポイントのうちの１つにおいて前記記述に従ってフォーマットされているクライアントメッセージを受信した際に、前記プロセッサに、
前記記述を使用して、前記メッセージからの命令を解析することと、
前記パッケージアプリケーションプログラミングインターフェースを介して、前記命令を前記複数のパッケージのうちの１つにパスすることと
を行わせる、コンピュータハードウェアシステム。
【請求項８】
前記サービス展開エンジンはさらに、実行されると、前記プロセッサに、
前記複数のパッケージのうちの前のバージョンのパッケージに基づいて、コンピュータハードウェアシステムが、全ての未処理動作を完了したときを検出することと、
前記前のバージョンのパッケージに基づいて、前記コンピュータハードウェアシステムが、新しい動作を容認することを妨げることと、
前記前のバージョンのパッケージを除去することと
を行わせる命令を備える、請求項７に記載のシステム。
【請求項９】
前記記述は、ジャグ文字列配列である、請求項７に記載のシステム。
【請求項１０】
前記記述は、インターフェース記述言語から形成される、請求項７に記載のシステム。
【請求項１１】
前記インターフェース記述言語は、ウェブサービス記述言語である、請求項１０に記載

のシステム。
【請求項１２】
前記クライアントメッセージは、プレーンテキストでフォーマットされている、請求項７に記載のシステム。
【請求項１３】
クライアントアプリケーションを実行する、複数のユーザデバイスと、
ネットワークを介して前記複数のユーザデバイスに接続される、企業サーバであって、前記サーバは、プロセッサと、メモリとを有し、前記メモリは、
複数のパッケージと、
コンピュータ実行可能命令を備えるサービス展開エンジンであって、前記コンピュータ実行可能命令は、実行されると、前記プロセッサに、サービスアプリケーションプログラミングインターフェースの記述を修正することなく、複数のパッケージを複数のエンドポイントに展開することを行わせる、サービス展開エンジンと、
命令のセットと
を備える、企業サーバと
を備えるネットワーク化システムであって、
各パッケージは、サービスを有し、一貫したパッケージアプリケーションプログラミングインターフェースを実装し、
前記企業サーバは、前記プロセッサによる前記命令のセットの実行を通して、前記クライアントアプリケーションに基づいて前記サービスにアクセスすることを可能にし、
前記命令のセットは、実行されると、前記プロセッサに、
前記サービスアプリケーションプログラミングインターフェースの前記記述に従ってフォーマットされている引数を含む、前記クライアントアプリケーションからのメッセージを解析することと、
前記メッセージに基づく命令を前記複数のパッケージのうちの１つにパスすることと
を行わせるコンピュータ実行可能命令である、ネットワーク化システム。
【請求項１４】
メモリに結合される、少なくとも１つの処理ユニットを備える、コンピュータハードウェアシステムであって、前記メモリは、コンピュータ実行可能命令で符号化されており、前記コンピュータ実行可能命令は、実行されると、前記少なくとも１つの処理ユニットに、
公開エンドポイントにおいて、クライアントメッセージを受信することであって、前記メッセージは、第１のアプリケーションプログラミングインターフェースの記述に従ってフォーマットされており、前記記述は、多次元配列を備える、ことと、
前記メッセージを解析することと、
前記メッセージのコンテンツをパッケージにパスすることであって、前記パッケージに基づいて、前記コンピュータハードウェアシステムは、前記パスされたコンテンツに関連付けられた動作を実行することが可能であり、前記パッケージは、第２のアプリケーションプログラミングインターフェースおよび論理を備え、前記コンピュータハードウェアシステムは、前記第２のアプリケーションプログラミングインターフェースを経由して、前記動作および前記論理に基づく結果を受信するように構成されている、ことと
を行わせる、システム。
【請求項１５】
前記記述は、インターフェース記述言語から形成される、請求項１４に記載のシステム。
【請求項１６】
前記少なくとも１つの処理ユニットはさらに、前記クライアントメッセージを認証させられる、請求項１４に記載のシステム。
【請求項１７】
前記パッケージは、複数のパッケージのうちの１つである、請求項１４に記載のシステ

ム。
【請求項１８】
前記複数のパッケージは、パッケージ自己割当バージョン番号に従って編成される、請求項１７に記載のシステム。
【請求項１９】
各パッケージの複数のバージョン番号が同時に存在する、請求項１８に記載のシステム。
【請求項２０】
前記公開エンドポイントは、複数の公開エンドポイントのうちの１つである、請求項１４に記載のシステム。
【請求項２１】
前記公開エンドポイントは、クライアントアクセス可能な場所を含み、前記クライアントアクセス可能な場所は、ポート、ユニフォームリソース識別子、名前付きトークン、名前付きパイプ、または共有メモリのブロックのうちの１つ以上を備える、請求項２０に記載のシステム。
【請求項２２】
異なるタイプのクライアント通信が、異なる公開エンドポイントにわたって同時に可能にされる、請求項２０に記載のシステム。
【請求項２３】
前記メッセージが有効な要求を含むかどうかを検出するステップをさらに含む、請求項１４に記載のシステム。
【請求項２４】
前記検出するステップは、前記メッセージの前記コンテンツが有効なパッケージにダイレクトされるかどうかを試験することを含む、請求項２３に記載のシステム。
【請求項２５】
前記第１のアプリケーションプログラミングインターフェースによって提供される抽象化の程度は、前記第２のアプリケーションプログラミングインターフェースによって提供される抽象化の程度よりも広い、請求項１４に記載のシステム。
【請求項２６】
プロセッサと、
複数の公開エンドポイントと、
メモリと
を備えるコンピュータハードウェアシステムであって、前記メモリは、
パッケージアプリケーションプログラミングインターフェースを実装する複数のパッケージと、
サービスアプリケーションプログラミングインターフェースの機能性の記述と、
コンピュータ実行可能命令を備えるサービス展開エンジンであって、前記コンピュータ実行可能命令は、実行されると、前記プロセッサに、前記サービスアプリケーションプログラミングインターフェースの前記機能性の前記記述を修正することなく、前記複数のパッケージを複数のエンドポイントに展開することを行わせる、サービス展開エンジンと
を備え、前記サービスアプリケーションプログラミングインターフェースは、コンピュータ実行可能命令を備え、前記コンピュータ実行可能命令は、実行されると、前記複数の公開エンドポイントのうちの１つにおいて前記記述に従ってフォーマットされているクライアントメッセージを受信した際に、前記プロセッサに、
前記記述を使用して、前記メッセージからの命令を解析することと、
前記命令に基づいて、前記クライアントメッセージに応答すること、前記クライアントメッセージが有効な要求を含むかどうかを検出すること、または、前記パッケージアプリケーションプログラミングインターフェースを介して、前記命令を前記複数のパッケージのうちの１つにパスすることのうちの１つ以上を含むアクションを行うことと
を行わせる、コンピュータハードウェアシステム。

【請求項２７】
前記サービス展開エンジンはさらに、実行されると、前記プロセッサに、
前記複数のパッケージのうちの前のバージョンのパッケージに基づいて、コンピュータハードウェアシステムが、全ての未処理動作を完了したときを検出することと、
前記前のバージョンのパッケージに基づいて、前記コンピュータハードウェアシステムが、新しい動作を容認することを妨げることと、
前記前のバージョンのパッケージを除去することと
を行わせる命令を備える、請求項２６に記載のシステム。
【請求項２８】
前記記述は、ジャグ文字列配列を含む、請求項２６に記載のシステム。
【請求項２９】
前記記述は、インターフェース記述言語から形成される、請求項２６に記載のシステム。
【請求項３０】
前記インターフェース記述言語は、ウェブサービス記述言語である、請求項２９に記載のシステム。
【請求項３１】
前記クライアントメッセージは、プレーンテキストでフォーマットされている、請求項２６に記載のシステム。
【請求項３２】
クライアントアプリケーションを実行する、複数のユーザデバイスと、
ネットワークを介して前記複数のユーザデバイスに接続される、企業サーバであって、前記サーバは、プロセッサと、メモリとを有し、前記メモリは、
複数のパッケージと、
コンピュータ実行可能命令を備えるサービス展開エンジンであって、前記コンピュータ実行可能命令は、実行されると、前記プロセッサに、サービスアプリケーションプログラミングインターフェースの記述を修正することなく、前記複数のパッケージを複数のエンドポイントに展開することを行わせる、サービス展開エンジンと、
命令のセットと
を備える、企業サーバと
を備えるネットワーク化システムであって、
各パッケージは、サービスを有し、一貫したパッケージアプリケーションプログラミングインターフェースを実装し、
前記企業サーバは、前記プロセッサによる前記命令のセットの実行を通して、前記クライアントアプリケーションに基づいて前記サービスにアクセスすることを可能にし、
前記命令のセットは、実行されると、前記プロセッサに、
前記サービスアプリケーションプログラミングインターフェースの前記記述に従ってフォーマットされている引数を含む、前記クライアントアプリケーションからのクライアントメッセージを解析することと、
前記クライアントメッセージに応答すること、前記クライアントメッセージが有効な要求を含むかどうかを検出すること、または、前記サービスアプリケーションプログラミングインターフェースを介して、前記クライアントメッセージを前記サービスにパスすることのうちの１つ以上を含むアクションを行うことと
を行わせるコンピュータ実行可能命令である、ネットワーク化システム。
【請求項３３】
前記クライアントメッセージに応答して、前記サービスは、リソースにアクセスすること、プロセスを行うこと、定義された形式に従って前記クライアントメッセージへの応答を作成すること、またはこれらの組み合わせを行うように構成されている、請求項３２に記載のネットワーク化システム。
【発明の詳細な説明】

【技術分野】
【０００１】
本開示の実施形態は、概して、ネットワークサービスに関し、より具体的には、ネットワークサービスの動的展開に関する。
【背景技術】
【０００２】
典型的企業システムでは、企業サーバは、ウェブサービスを介したクライアントアプリケーションからの要求に応答して、バックエンドリソース（例えば、アプリケーション、データサーバ、またはサービスプロバイダ）に接続する。典型的には、ウェブサービスは、クライアントが、外部呼び出し元への離散要求としてではなく、遠隔プロシージャ呼び出しとしてサービスを扱うことを可能にするような方法で、企業サーバに公開される。これは、部分的に、殆どの統合開発環境が、クライアントアプリケーションの開発者がそのようなサービスをアプリケーションに容易に組み込むことを可能にするため、行われる。開発者にとって容易であるが、本行為は、アプリケーションと遠隔サービスとの間に隠れたハードリンクを導入する。ハードリンクは、具体的記述において記憶されるアプリケーションとサービスとの間の一種の契約として表され得る。本契約は、ある状況では有益であり得るが、サービスまたはクライアントへの変更は、デバイスとブレイクリンクとの間で保持される契約を無効にし得る。これは、絶え間なく変わる企業環境において問題である、バグが多い互換性のないソフトウェアおよび不良なユーザ体験をもたらし得る。ハードリンクを修復または更新することは、時間およびリソースを要し得、サーバによって提供されるサービスの中断をもたらし得る。したがって、当技術分野では、遠隔サービス契約を無効にする危険性を低減させる、ロバストなクライアントサーバ能力を提供する必要性がある。
【発明の概要】
【課題を解決するための手段】
【０００３】
ある実装は、メモリに結合される、少なくとも１つの処理ユニットを備える、コンピュータシステムを含んでもよく、メモリは、コンピュータ実行可能命令で符号化され、該命令は、実行されると、少なくとも１つの処理ユニットに、公開エンドポイントにおいて、クライアントメッセージを受信することであって、該メッセージは、第１のアプリケーションプログラミングインターフェースの第１の記述に従ってフォーマットされている、ことと、第２の記述に従ってメッセージからの命令を解析することと、第１のアプリケーションプログラミングインターフェースの機能を介して、パッケージに命令をパスすることであって、該パッケージは、第２のアプリケーションプログラミングインターフェースおよび論理を有する、ことと、第２のアプリケーションプログラミングインターフェースを経由して、命令および論理に基づく結果を受信することと、結果に基づいてクライアントメッセージに応答することとを行わせる。第１の記述は、例えば、ジャグ文字列配列の形態で、実質的に一般的な形式を有してもよい。加えて、第１のアプリケーションプログラミングインターフェースは、第２のアプリケーションプログラミングインターフェースより実質的に一般的であり得る。第１および第２の記述は、インターフェース記述言語（例えば、ウェブサービス記述言語）から形成される。クライアントメッセージは、プレーンテキストでフォーマットされてもよい。
【０００４】
加えて、または代替として、実装は、プロセッサと、複数の公開エンドポイントと、メモリとを備える、コンピュータハードウェアシステムを含んでもよく、該メモリは、パッケージアプリケーションプログラミングインターフェースと、サービスアプリケーションプログラミングインターフェースの機能性の記述と、コンピュータ実行可能命令を備えるサービス展開エンジンであって、該コンピュータ実行可能命令は、実行されると、プロセッサに、記述を実質的に修正することなく、各パッケージがパッケージアプリケーションプログラミングインターフェースを実装する、複数のパッケージを複数のエンドポイント

に展開させる、サービス展開エンジンと、サービスアプリケーションプログラミングインターフェースであって、実行されると、複数の公開エンドポイントのうちの１つにおいて記述に従ってフォーマットされているクライアントメッセージを受信した際に、プロセッサに、記述を使用して、メッセージからの命令を解析することと、パッケージアプリケーションプログラミングインターフェースを介して、命令を複数のパッケージのうちの１つにパスすることとを行わせるコンピュータ実行可能命令を備える、サービスアプリケーションプログラミングインターフェースとを備える。サービス展開エンジンはさらに、実行されると、プロセッサに、複数のパッケージのうちの前のバージョンのパッケージが全ての顕著な動作を完了したときを検出することと、前のバージョンのパッケージが新しい動作を容認することを妨げることと、前のバージョンのパッケージを除去することとを行わせる命令を備えてもよい。実質的に一般的な形式は、例えば、ジャグ文字列配列であってもよい。第１および第２の記述は、ウェブサービス記述言語等のインターフェース記述言語から形成されてもよい。クライアントメッセージは、プレーンテキストでフォーマットされてもよい。

【０００５】

加えて、または代替として、実装は、クライアントアプリケーションを実行する、複数のユーザデバイスと、ネットワークを介して複数のクライアントデバイスに接続される、企業サーバとを備える、ネットワーク化システムを含んでもよい。サーバは、プロセッサと、複数のパッケージおよび命令のセットを備える、メモリとを有してもよい。各パッケージは、サービスを有し、同一のパッケージアプリケーションプログラミングインターフェースを実装してもよい。企業サーバは、プロセッサによる命令のセットの実行を通して、クライアントアプリケーションがサービスにアクセスすることを可能にしてもよい。命令のセットは、実行されると、少なくとも１つのプロセッサに、単純アプリケーションプログラミングインターフェースの一般的記述に従ってフォーマットされている引数を含有する、クライアントアプリケーションからのメッセージを解析させ、命令を複数のパッケージのうちの１つにパスさせる、コンピュータ実行可能命令であってもよい。

本発明は、例えば、以下を提供する。

（項目１）

メモリに結合される、少なくとも１つの処理ユニットを備える、コンピュータハードウェアシステムであって、前記メモリは、コンピュータ実行可能命令で符号化されており、前記コンピュータ実行可能命令は、実行されると、前記少なくとも１つの処理ユニットに、

公開エンドポイントにおいて、クライアントメッセージを受信することであって、前記メッセージは、第１のアプリケーションプログラミングインターフェースの第１の記述に従ってフォーマットされている、ことと、

第２の記述に従って前記メッセージからの命令を解析することと、

前記第１のアプリケーションプログラミングインターフェースの機能を介して、パッケージに前記命令をパスすることであって、前記パッケージは、第２のアプリケーションプログラミングインターフェースおよび論理を有する、ことと、

前記第２のアプリケーションプログラミングインターフェースを経由して、前記命令および前記論理に基づく結果を受信することと、

前記結果に基づいて前記クライアントメッセージに応答することと

を行わせ、

前記第１の記述は、実質的に一般的な形式を有する、システム。

（項目２）

前記第１のアプリケーションプログラミングインターフェースは、前記第２のアプリケーションプログラミングインターフェースより実質的に一般的である、項目１に記載のシステム。

（項目３）

前記実質的に一般的な形式は、ジャグ文字列配列である、項目１に記載のシステム。

（項目４）
前記第１および第２の記述は、インターフェース記述言語から形成される、項目１に記載のシステム。
（項目５）
前記インターフェース記述言語は、ウェブサービス記述言語である、項目４に記載のシステム。
（項目６）
前記クライアントメッセージは、プレーンテキストでフォーマットされている、項目１に記載のシステム。
（項目７）
プロセッサと、
複数の公開エンドポイントと、
メモリであって、
パッケージアプリケーションプログラミングインターフェースと、
サービスアプリケーションプログラミングインターフェースの機能性の記述と、
コンピュータ実行可能命令を備えるサービス展開エンジンであって、前記コンピュータ実行可能命令は、実行されると、前記プロセッサに、前記記述を実質的に修正することなく、複数のパッケージを複数のエンドポイントに展開させ、各パッケージは、前記パッケージアプリケーションプログラミングインターフェースを実装する、サービス展開エンジンと、
コンピュータ実行可能命令を備えるサービスアプリケーションプログラミングインターフェースであって、前記コンピュータ実行可能命令は、実行されると、前記複数の公開エンドポイントのうちの１つにおいて前記記述に従ってフォーマットされているクライアントメッセージを受信した際に、前記プロセッサに、
前記記述を使用して、前記メッセージからの命令を解析することと、
前記パッケージアプリケーションプログラミングインターフェースを介して、前記命令を前記複数のパッケージのうちの１つにパスすることと
を行わせる、サービスアプリケーションプログラミングインターフェースと
を備える、メモリと
を備える、コンピュータハードウェアシステム。
（項目８）
前記サービス展開エンジンはさらに、実行されると、前記プロセッサに、
前記複数のパッケージのうちの前のバージョンのパッケージが全ての未処理動作を完了したときを検出することと、
前記前のバージョンのパッケージが新しい動作を容認することを妨げることと、
前記前のバージョンのパッケージを除去することと
を行わせる命令を備える、項目７に記載のシステム。
（項目９）
前記実質的に一般的な形式は、ジャグ文字列配列である、項目８に記載のシステム。
（項目１０）
前記第１および第２の記述は、インターフェース記述言語から形成される、項目８に記載のシステム。
（項目１１）
前記インターフェース記述言語は、ウェブサービス記述言語である、項目１０に記載のシステム。
（項目１２）
前記クライアントメッセージは、プレーンテキストでフォーマットされている、項目８に記載のシステム。
（項目１３）
クライアントアプリケーションを実行する、複数のユーザデバイスと、

ネットワークを介して前記複数のクライアントデバイスに接続される、企業サーバであって、プロセッサと、複数のパッケージおよび命令のセットを備える、メモリとを有する、サーバと
を備え、
各パッケージは、サービスを有し、同一のパッケージアプリケーションプログラミングインターフェースを実装し、
前記企業サーバは、前記プロセッサによる前記命令のセットの実行を通して、前記クライアントアプリケーションが前記サービスにアクセスすることを可能にし、
前記命令のセットは、実行されると、前記少なくとも１つのプロセッサに、
単純アプリケーションプログラミングインターフェースの一般的記述に従ってフォーマットされている引数を含有する、前記クライアントアプリケーションからのメッセージを解析することと、
前記命令を前記複数のパッケージのうちの１つにパスすることと
を行わせるコンピュータ実行可能命令である、ネットワーク化システム。

【図面の簡単な説明】

【０００６】

【図１】図１は、ネットワーク化クライアントアプリケーションおよび遠隔サービスのある実装の概略ブロック図を図示する。

【図２】図２は、多次元配列の形態で一般的形式のある実装を図示する。

【図３】図３は、コンピュータネットワーキング環境のある実装の概略ブロック図を図示する。

【図４】図４は、ある実装による、コンピュータ可読媒体上に位置する特定のモジュールの概略ブロック図を図示する。

【図５】図５は、ある実装による、要求を処理するための方法のフローチャートである。

【図６】図６は、ある実装による、パッケージに作用するための方法のフローチャートである。

【発明を実施するための形態】

【０００７】

動的サービス展開のためのシステムおよび方法が、本明細書に開示される。ある詳細が、本開示の実施形態の十分な理解を提供するために以下に記載される。しかしながら、本開示の実施形態は、これらの特定の詳細を伴わずに実装されてもよい。また、特定の実施形態は、一例として提供され、限定的として解釈されるべきではない。他の場合において、周知の回路、制御信号、タイミングプロトコル、およびソフトウェア動作は、本発明を不必要に曖昧にすることを回避するために詳細に示されていない。

【０００８】

開示される実施形態は、概して、ネットワークを経由してサーバによってクライアントに提供されるサービスに関する。例えば、ユーザは、所望の結果を生じるために、サービスと相互作用してサーバのリソースまたは機能性にアクセスするようにクライアントに指図してもよい。サーバは、多くの場合、種々のハードウェアおよびソフトウェアアーキテクチャのクライアントと相互作用するように構成され、これは、互換性の懸念を導入する。したがって、サーバは、異なるクライアントアーキテクチャとの優れた互換性を達成するために、種々の抽象化の層を定義してもよい。しかしながら、抽象化層が柔軟または抽象的すぎる場合、クライアントおよびサーバは、要求およびフォーマットの一貫した理解を確実にすることの困難に遭遇し得る。したがって、相互定義およびフォーマットが有益であり得る。これは、インターフェース記述言語（ＩＤＬ）の記述に従ってフォーマットされている配信プロトコルの使用を通して達成されてもよい。

【０００９】

ＩＤＬは、サーバによって提供されるウェブサービス等のサービスによって提供される機能性を記述するために使用される、言語またはフォーマットであってもよい。具体的には、ＩＤＬは、多くの場合、ＩＤＬ記述を含有するファイルを介して、サービスが呼び出

され得る様式、サービスによって予期されるパラメータ、呼び出しに応答してサービスによって提供される応答のタイプ（例えば、返されるデータ構造のタイプ）、および／または他の機能性を規定するために使用されてもよい。ＩＤＬは、独自の一意の言語、既知または既存の言語（例えば、拡張マークアップ言語（ＸＭＬ））で記述をフォーマットもしくは作成する方法、またはそれらの組み合わせであってもよい。ＩＤＬの実施例は、ウェブサービス記述言語（ＷＳＤＬ）、ウェブアプリケーション記述言語、およびＡｎｄｒｏｉｄ（ＴＭ）インターフェース定義言語を含んでもよい。

【００１０】

ある場合に、ＩＤＬ記述は、データがネットワークを経由して２つまたはそれを上回るアプリケーションの間で交換される様式を規定する、１つまたはそれを上回る配信プロトコルと組み合わせて使用されてもよい。各プロトコルは、例えば、構造化メッセージングフレームワークを提供してもよい。これらのプロトコルは、シンプルオブジェクトアクセスプロトコル（ＳＯＡＰ）および表現状態転送（ＲＥＳＴ）を含んでもよいが、それらに限定されない。ひいては、これらの配信機構は、ハイパーテキストトランスポートプロトコル（ＨＴＴＰ）またはシンプルメールトランスポートプロトコル（ＳＭＴＰ）等の１つまたはそれを上回るアプリケーション層トランスポートプロトコルに依拠してもよい。

【００１１】

図１は、ネットワーク１１０を経由して遠隔サービス７０と通信するように構成されるアプリケーション５０を含む、システム１０の実施形態を図示する。実施例として、アプリケーション５０は、ユーザコンピュータデバイス上で実行する命令を備えてもよく、遠隔サービス７０は、企業サーバ上で実行する命令を備えてもよい。ある実装では、アプリケーション５０は、３つの主要な機能を有してもよく、すなわち、ユーザインターフェースをユーザに提供すること、遠隔サービス７０へのメッセージ９０（例えば、呼び出し）を作成してフォーマットすること、そして、遠隔サービス７０から応答を受信して処理することである。ユーザインターフェースは、出力をユーザに提供し、ユーザから入力を受信する方法であってもよい。本入力および出力は、遠隔サービス７０への要求および遠隔サービス７０からの応答に関係付けられてもよい。

【００１２】

ある実装では、アプリケーション５０は、クライアントアプリケーション５０に既知またはアクセス可能であるＩＤＬ記述６０に従って、メッセージ９０をフォーマットし、ネットワーク１１０を経由して遠隔サービス７０に提供してもよい。遠隔サービス７０は、ＩＤＬ記述６０と互換性があるＩＤＬ記述８０を含んでもよい（例えば、ＩＤＬ記述６０およびＩＤＬ記述８０は、同一である）。遠隔サービス７０は、プロトコルおよびＩＤＬ記述８０に従って、メッセージ９０を受信し、メッセージ９０を解析してもよい。

【００１３】

メッセージ９０は、種々の方法でフォーマットされてもよい。例えば、メッセージ９０は、プレーンテキストとしてフォーマットされてもよい。プレーンテキストは、有意な処理またはフォーマットが欠けている形式として表されてもよい。例えば、プレーンテキストは、ＡＳＣＩＩ、ＵＴＦ－８、または同等物に従って符号化されてもよい。プレーンテキストは、例えば、画像および符号化された番号で行われるように、バイナリオブジェクトとして解釈することを必要としない。

【００１４】

メッセージ９０のコンテンツに応じて、遠隔サービス７０は、特定のリソースにアクセスすること、プロセスを行うこと、および同意した形式に従ってクライアントアプリケーション５０への応答を作成することを含むが、それらに限定されない、ある行動をとってもよい。このようにして、アプリケーション５０および遠隔サービス７０は、補完的ＩＤＬ記述６０、８０によって定義される遠隔サービス契約に従って動作してもよい。

【００１５】

しかしながら、ある場合に、遠隔サービス７０またはクライアントアプリケーション５０は、修正（例えば、アップグレード）されてもよい。ＩＤＬ記述６０、８０が、特に具

体的な様式でフォーマットされた場合には、ＩＤＬ記述６０、８０は、もはや提供されるサービスを正確に記述しなくなり得る。結果として、アプリケーション５０およびサービス７０が、互換性のあるＩＤＬ記述６０、８０に従ってメッセージ９０をフォーマットして解析することができない場合があるため、アプリケーション５０と遠隔サービス７０との間の契約が無効にされ得る。非互換性は、アプリケーション５０にもはや有効な要求を遠隔サービス７０に提供させなくし、有効な応答を受信させなくし得る。互換性のないＩＤＬ記述６０、８０に従った通信は、誤ったデータ、アプリケーション不安定性、または他の問題をもたらし得る。しかしながら、一般的ＩＤＬ記述６０、８０は、基礎的サービスへの修正および／またはアップグレードにもかかわらず、契約が完全なままであり得るように実装され得る。ＩＤＬ記述６０、８０は、種々のサービスのためのエントリポイントとしての機能を果たすように、一般的形式で書かれてもよい。一般的形式のある実装は、文字列の多次元配列を含むが、それに限定されない、データ構造の使用を含んでもよい。

【００１６】

図２は、多次元配列２１０の形態で一般的形式のある実装を図示する。多次元配列２１０は、他の配列２５０、２５１への参照２４０、２４１を含有し得る、要素２３０、２３１、２３２を含む。他の配列２５０、２５１はまた、それら自体が要素を定義してもよい。これらの要素は、さらに他のデータ構造またはデータへの参照を備えてもよい。ある図示される実装では、他の配列２５０、２５１は、多次元配列２１０をジャグ文字列配列にする、文字データを含有してもよい。要素２３２は、別の配列への参照を含有せず、ヌル参照を有するものとして表されてもよい。多次元配列は、固定長を有する、可変長を有する、特定のタイプ（例えば、文字列）に嵌められる、および多次元配列で典型的に見出される他の属性を含むが、それらに限定されない、種々の性質を有してもよい。

【００１７】

図３は、ユーザデバイス１０２と、ネットワーク１１０と、企業サーバ１２０とを含む、コンピュータネットワーキング環境１００のある実装の概略図を図示する。ユーザデバイス１０２は、モデム、ルータ、ゲートウェイ、サーバ、シンクライアント、ラップトップ、デスクトップ、コンピュータ、タブレット、メディアデバイス、スマートフォン、テレビ、ファブレット、携帯電話、または他のモバイルデバイス、もしくは同デバイスの任意の組み合わせまたは副次的組み合わせを含むが、それらに限定されない、コンピュータデバイスを備えてもよい。ユーザデバイス１０２は、アプリケーション５０の実行を可能にする機能性を提供するように、ユーザデバイス１０２の１つまたはそれを上回る処理ユニット６４と併せて動作し得る、実行可能命令で符号化されたコンピュータ可読媒体６２を含んでもよい。コンピュータ可読媒体６２はまた、ＩＤＬ記述６０を含んでもよい。アプリケーション５０は、企業サーバ１２０によって提供される１つまたはそれを上回るサービスとインターフェースをとり得る、実行可能プログラム等のアプリケーションであってもよい。ユーザデバイス１０２は、ネットワーク１１０を経由して、企業サーバ１２０を含むがそれに限定されない、任意の数のデバイスと通信するように構成されてもよい。

【００１８】

ネットワーク１１０は、ローカルエリアネットワーク（ＬＡＮ）、広域ネットワーク（ＷＡＮ）、メトロポリタンエリアネットワーク（ＭＡＮ）、セルラーネットワーク、および／またはインターネット等の１つもしくはそれを上回るネットワークを備えてもよい。ネットワーク１１０に、ネットワーク１１０から、ならびにネットワーク１１０内で提供される通信は、有線および／または無線であり得、さらに、現在もしくは将来当技術分野で公知である、任意のネットワーキングデバイスによって提供されてもよい。ネットワーク１１０を経由して通信するデバイスは、伝送制御プロトコル／インターネットプロトコル（ＴＣＰ／ＩＰ）またはユーザデータグラムプロトコル（ＵＤＰ）等の通信プロトコルを用いて通信してもよい。加えて、ユーザデバイス１０２および企業サーバ１２０は、ハイパーテキスト転送プロトコル（ＨＴＴＰ）、ハイパーテキスト転送プロトコルセキュア（ＨＴＴＰＳ）、セキュアソケット層（ＳＳＬ）、サーバ常駐プロトコル、または他のプ

ロトコル等の公知のプロトコルを使用して、通信するように構成されてもよい。サーバ常駐プロトコルは、名前付きパイプ、共有メモリ、および他のプロトコルを含んでもよい。そのようなプロトコルはまた、同一の物理的ユニット内で、アプリケーションサーバ（例えば、管理する、バックエンドプロセスを実行する、またはアプリケーションをホストするサーバ）と企業サーバ１２０との間で情報を共有するために使用されてもよい。

【００１９】

企業サーバ１２０は、１つまたはそれを上回るコンピュータ可読媒体１２３に動作可能に結合される、１つまたはそれを上回る処理ユニット１２１を含んでもよい。コンピュータ可読媒体１２３は、外部もしくは内部に取り付けられたハードディスクドライブ、ソリッドステート記憶装置（ＮＡＮＤフラッシュまたはＮＯＲフラッシュメディア等）、階層記憶ソリューション、記憶エリアネットワーク、ネットワークアタッチド記憶装置、および／または光学記憶装置を含むが、それらに限定されない、一過性または非一過性である、任意の形態のコンピュータ可読記憶装置もしくはコンピュータ可読メモリを含んでもよい。

【００２０】

コンピュータ可読媒体１２３は、サービス展開エンジン５２７、ＩＤＬ記述８０、ならびに複数のパッケージおよびサービス５２５を操作するための実行可能命令を含むが、それらに限定されない、種々のモジュールを記憶してもよい。説明されるように、コンピュータ可読媒体１２３上に記憶された命令は、１つまたはそれを上回る処理ユニット１２１もしくは他の処理ユニット上で実行されてもよい。

【００２１】

サービス展開エンジン５２７を操作するための実行可能命令は、処理ユニット１２１上で実行されると、リアルタイムでパッケージならびにサービス５２５を調査、展開、開始／停止、および／またはアンインストールするために動的ライブラリローダの役割を果たすように、企業サーバ１２０上のサービス展開５２７の動作を可能にする、命令を含んでもよく、そのさらなる実施例は、以下で提供される。サービス展開エンジン５２７のための実行可能命令が、同一のコンピュータ可読媒体１２３上で示されるが、いくつかの実施形態では、命令のいずれかまたは全てのセットが、複数のコンピュータ可読媒体上に提供されてもよく、同一の媒体上に常駐しなくてもよい。

【００２２】

パッケージおよびサービス５２５は、企業サーバ１２０の処理ユニット１２１上で実行されたときに特定の機能性もしくはサービスを提供するためにともにパッケージ化されている、１つまたはそれを上回るソフトウェア構成要素およびリソースであってもよい。図４を参照すると、パッケージおよびサービス５２５は、複数の個々のパッケージ５３０（例えば、５３０Ａ－５３０Ｎ）を備えてもよい。パッケージ５３０は、パッケージ自己割当バージョン番号に従って編成され、同一パッケージの複数のバージョンが同時に存在することを可能にしてもよい。特に、大企業に関して、本内蔵バージョニング機能性は、重要なパッケージ５３０の新しいバージョンへのスケーリングを可能にするが、カットオーバアプローチを必要とせず、後者は、多くの場合、隠されたトラップおよびサービス中断の危険を伴う。同一パッケージ５３０の複数のバージョンが同時に存在することを可能にすることは、２つのバージョンが存在し、異なるユーザデバイス１０２によって企業サーバ１２０上で同時に使用されることを可能にする。

【００２３】

続けて図４を参照すると、本図は、コンピュータ可読媒体１２３上に位置する特定のモジュールの詳細な図を図示し、サービスアプリケーションプログラミングインターフェース（ＡＰＩ）５２０と、パッケージおよびサービス５２５と、公開エンドポイント５２６（例えば、５２６Ａ－５２６Ｎ）と、サービス展開エンジン５２７とを含む。

【００２４】

公開エンドポイント５２６は、クライアントアプリケーションが企業サーバ１２０によって実装される機能（例えば、ＡＰＩ）にアクセスし得る、企業サーバ１２０によってエ

クスポーズされる場所（例えば、ポート、ユニフォームリソースリンク（ＵＲＬ）等のユニフォームリソース識別子（ＵＲＩ）、名前付きトークン、名前付きパイプ、共有メモリのブロック、または他の場所）であってもよい。異なるタイプの通信が、異なるエンドポイントを経由して可能にされてもよい。例えば、ＴＣＰ／ＩＰエンドポイント、ＳＯＡＰエンドポイント等があってもよい。種々のサービスへの呼び出し中に、具体的タイプのエンドポイント５２６（例えば、Ｗｉｎｄｏｗｓ（登録商標） Ｃｏｍｍｕｎｉｃａｔｉｏｎ Ｆｏｕｎｄａｔｉｏｎ（ＷＣＦ）、ＪａｖａＳｃｒｉｐｔ（登録商標） Ｏｂｊｅｃｔ Ｎｏｔａｔｉｏｎ（ＪＳＯＮ）、ＳＯＡＰ、ＲＥＳＴ、ＴＣＰ／ＩＰ、名前付きパイプ）が、メッセージ９０を送信するアプリケーション５０の通信能力によって選定されてもよい。企業サーバ１２０は、同時に全ての公開エンドポイント５２６上で独自のＩＤＬ記述８０を公開し、同様に、同時にこれらの全てへの実況変更（すなわち、ゼロダウンタイム）を公開してもよい。これは、パッケージ５３０が企業サーバ１２０の中へ展開されるときに、所望のエンドポイント５２６のうちのありとあらゆるものを介して、同時かつ瞬時に利用可能および到達可能であることを意味する。

【００２５】

サービスＡＰＩ５２０は、行われることになる基本的機能のセット（例えば、作成、読取、更新、削除、および起動）を有効にする、単純インターフェースであってもよい。ＩＤＬ記述８０は、サービスＡＰＩ５２０の機能性を記述してもよい。例えば、ＩＤＬ記述６０、８０は、クライアントアプリケーション５０が、アドレス方式に従って具体的機能を実行するように企業サーバ１２０に命令するメッセージ９０を送信することを可能にする、一般的な「ＩｎｖｏｋｅＯｐ」機能を記述してもよい。例えば、企業「ｃｏｍｐａｎｙ」からのパッケージ「ｐａｃｋａｇｅ」のバージョン２．０から動作「ｏｐｅｒａｔｉｏｎ」を実行し得る、「ＩｎｖｏｋｅＯｐ（「ｃｏｍｐａｎｙ．ｐａｃｋａｇｅ［２０］．ｏｐｅｒａｔｉｏｎ」）である。企業サーバ１２０は、代わりに呼び出し元が最新バージョンを好むであろう場合に、常に、企業サーバ１２０内の内部でインストールされたパッケージの最新バージョンに決定する、バージョン［＊］（すなわち、ワイルドカードバージョン）を単純に要求する必要があるように構成されてもよい。企業サーバ１２０は、いかなるバージョンも規定されない場合、アプリケーション５０が要求された動作を行うように最新または最高バージョンのパッケージを単純に好むであろうことを仮定するように、構成されてもよい。

【００２６】

サービスＡＰＩ５２０は、同一のＩＤＬ記述８０を全てのクライアントアプリケーション５０にエクスポーズし、企業サーバ１２０自体の機能として動作ダイナミズムを実装してもよい。本構造は、クライアントアプリケーション５０および企業サーバ１２０がＩＤＬ記述６０、８０の間で果たされる契約上の合意を保持することを可能にする。企業サーバ１２０は、クライアントアプリケーション５０と企業サーバ１２０との間の通信機構が、企業サーバ１２０とクライアントアプリケーション５０との間のＩＤＬ記述契約を無効にするような方法で変化しないであろうことを保証するために、一般的形式（例えば、配列の配列、または図２に関連して説明されるジャグ文字列配列）を利用する、公開パッケージおよびサービス５２５への単純エントリポイント（例えば、１つだけのエントリポイント）を用いて、単純ＩＤＬ記述をエクスポーズしてもよい。複数のメッセージにわたって一定のままであるＩＤＬ記述８０に依存する、クライアントアプリケーション５０は、そのようにすることができ、企業サーバ１２０自体にエクスポーズされた表面の下で行われる、展開／展開解除動作による影響を受けないであろう。

【００２７】

図５は、ある実装による、メッセージ９０を処理するための方法５０００のフローチャートを図示する。ステップ５１００では、企業サーバ１２０が、公開エンドポイント５２６においてメッセージを受信する。次に、ステップ５２００では、予備処理が、メッセージに行われてもよい。本ステップ５２００は、付加的パケットを待つこと、パケットに誤差訂正を行うこと、予備パケット解析、および他のプロセスを含む、メッセージ９０を受

信することと関連付けられる、ネットワーキングタスクを取り扱うことを含んでもよい。加えて、企業サーバ１２０は、メッセージが認証されたユーザからであることを確実にすること、または呼び出しアプリケーション５０がそれ自体またはユーザを識別するために正規セッション識別子を使用していることを確実にすること等の認証ステップを含んでもよい。次に、ステップ５３００では、メッセージ９０が、ＩＤＬ記述８０に従って解析される。メッセージ９０を解析した後に、ステップ５４００では、企業サーバ１２０が、メッセージ９０をパッケージ５３０にパスすること、メッセージ９０の特定のコンテンツを特定のパッケージ５３０にパスすること、コンテンツをパッケージ５３０にパスすることなくメッセージ９０に応答すること、およびメッセージ９０が有効な要求を含有するかどうかを検出すること（例えば、コンテンツが有効なパッケージにダイレクトされるかどうかを試験すること）を含むが、それらに限定されない、メッセージ９０のコンテンツに基づく動作を行う。ある実装では、メッセージをパスすることは、特定のパッケージ５３０によって実装されるパッケージＡＰＩ５３２に従って行われてもよい。加えて、ある実装によると、いったんサービスＡＰＩ５２０が呼び出しを受信すると、サービスＡＰＩ５２０は、呼び出し要求オブジェクトに変換を行わない。本プロセスは、メッセージを受信することとそれに作用することとの間の待ち時間を短縮し得る。

【００２８】

図４に戻ると、企業サーバ１２０は、ＩＤＬ記述８０を変更する必要なく、個々のパッケージ５３０およびそれらの動作５３４へのアクセスを提供するために、独自のアドレス命名法を使用してもよい。ＩＤＬ記述８０は、動作を起動するための一般的方法を記述またはエクスポーズしてもよい。引数は、パラメータを規定するために変化してもよい。構文は、バージョン付きおよび／またはバージョンなし呼び出しをサポートするために拡張されてもよい。ある実装は、付加的パラメータをＩＤＬに追加することなく、そうしてもよい。代わりに、ある実装は、例えば、アドレス命令法自体に所望のバージョン番号を含むことによって、具体的バージョンがアドレス命令法自体に組み込まれることを可能にするように、動作を定義してもよい。

【００２９】

ＩＤＬ非依存性バージョン呼び出しへの本アプローチは、いくつかの利点を提示する。例えば、ユーザデバイス１０２は、最高バージョン番号を要求し、精度または誤差について結果を試験するようにプログラムされてもよい。結果が誤差を含有する場合には、クライアントは、正確な／誤差を含まない結果を受信する、または呼び出すバージョン番号がなくなるまで、一連の増分的に低いバージョン番号を要求し得る。これは、例えば、発見されていない誤差がパッケージのより新しいバージョンに存在する場合に有用であり得る。加えて、本システムは、動作の異なるバージョンにわたって機能を利用するようにクライアントアプリケーションを構成するために、使用され得る。例えば、ユーザは、アプリケーションの１つのバージョン（例えば、バージョン１．０）で見出されるいくつかの特徴を好み得るが、別のバージョン（例えば、バージョン２．０）で見出される他の特徴を好む。ユーザまたはアプリケーションは、いくつかの特徴にバージョン１．０を利用すること、および全ての他の特徴にバージョン２．０を利用することを規定してもよい。加えて、パッケージの別個の性質は、異なるバージョンのユーザを妨害することなく、必要に応じて何度も、特定のバージョンが再び呼び出されること、または中断されることを可能にする。

【００３０】

加えて、マスタサービスに他のサービスを包み、全ての呼び出しアプリケーションが、個々のサービス自体への脆弱なリンク機構に依存する代わりに、本ラッパサービスを使用することを要求する、カスタムソリューションが作成され得る。しかしながら、マスタサービスが依存性サービス中断から現在の呼び出し元を隔離する可能性が低くあり得るため、本アプローチは、オーバーヘッドを導入し得る。これはまた、ラッパ／マスタサービスとその依存関係との間のリンクが、ラッパと包まれたサービスとの間のネットワーク上の負荷を増加させ得る、接続間呼び出しのウェブを導入するため、時間集約的であり得る。

【００３１】
続けて図４を参照すると、ある実装では、各パッケージ５３０は、パッケージＡＰＩ５３２、ならびに動作およびビジネス論理５３４を実装してもよい。パッケージＡＰＩ５３２は、特定のインターフェースであり、それを通してパッケージ５３０および企業サーバ１２０が通信し、パッケージ５３０の基礎的動作およびビジネス論理５３４へのアクセスを提供するステップを含み得る。パッケージＡＰＩ５３２ならびに動作およびビジネス論理５３４はまた、企業サーバ１２０が遠隔サービス７０を提供することも可能にし得る。したがって、企業サーバ１２０は、パッケージＡＰＩ５３２を介してメッセージ９０から動作およびビジネス論理５３４に解析された引数をパスするように構成されてもよい。
【００３２】
ある実装では、企業サーバ１２０は、一般的形式（例えば、単純な文字列の文字列）呼び出しパターンと相互作用することが可能な任意の公開エンドポイント５２６にわたって、任意のパッケージおよび任意の動作を同時に展開してもよい。これは、例えば、外部で一貫した単純サービスＡＰＩ５２０をエクスポーズし、企業サーバ１２０に展開するためにパッケージ開発者が使用しなければならない一貫したパッケージＡＰＩ５３２を内部で活用することによって、達成されてもよい。一貫したパッケージＡＰＩ５３２は、企業サーバ１２０と通信するために種々のプロトコルを使用して、クライアントアプリケーション５０へのエクスポージャを可能にする。
【００３３】
本機能性は、例えば、企業サーバ１２０上の認識されたパッケージ５３０としてビジネス論理５３４をインストールするために、企業サーバ１２０の内部で一貫したパッケージＡＰＩ５３２を利用することによって、達成されてもよい。パッケージＡＰＩ５３２は、これらの変化が呼び出しアプリケーション５０によって見られる、または感じられることを可能にすることなく、企業サーバ１２０が動的に調節し、種々のサービス要求に応答することを可能にする、内部で一貫した抽象化層を提供してもよい。具体的には、パッケージ５３０は、ＩＤＬ記述８０を変更する必要なく、展開および展開解除されてもよい。サービスＡＰＩ５２０によって提供される抽象化は、パッケージおよびサービス５２５への変更にもかかわらず、ＩＤＬ記述８０が一貫したままであり得る（例えば、パッケージ５３０が展開される、再展開される、または展開解除される）ように、パッケージＡＰＩ５３２よりも実質的に広く、より一般的、および／またはより柔軟であり得る。本配列は、パッケージＡＰＩ５３２および他の基礎的機能性の「それがどのようにして行われているか」から、ＩＤＬ記述８０によって記述される「するべきこと」を隔離することによって、パッケージＡＰＩ５３２がＩＤＬ記述８０から独立して動作することを可能にし得る。ある実装では、パッケージＡＰＩ５３２は、ＩＤＬ記述機能性とパラレルである動的登録概念（例えば、動作参照機能性、動作起動、および他の機能）を実装することによって、リンクを破損する危険性を伴わずにバイナリレベルで固定契約を有効にする。
【００３４】
システム１００が、一貫した内部パッケージＡＰＩ５３２を利用し、（例えば、サービスＡＰＩ５２０によって提供される抽象化を伴わずに）これらのパッケージ５３０を呼び出しクライアントアプリケーション５０に完全にはエクスポーズしないため、システム１００は、同時に多くのクライアントルートを通してパッケージ５３０によって提供される動作およびサービスをエクスポーズすることができる。例えば、動作５３４を伴うインストールされたパッケージ５３０は、自動的かつ同時に公開され、ＷＣＦ、ＳＯＡＰ、ＪＳＯＮ、ＲＥＳＴ、名前付きパイプ、ＴＣＰ／ＩＰ、およびその他等の複数のエンドポイント５２６を介して、企業に利用可能にされてもよい。企業サーバ１２０およびパッケージ５３２が両方とも、同一の専用かつ周知のパッケージＡＰＩ５３２を使用する（サーバ１２０がそれを消費または予期し、パッケージ開発者がそれを消費または実装する）ため、パッケージ５３０は、企業サーバ１２０自体への動的リアルタイム拡張の役割を果たし得る。インストールされたパッケージ５３０もまた、公開され、他のパッケージ５３０に利用可能にされてもよい。本広域公開機能性は、パッケージ開発者が余分なステップを行う

、それを要求する、または任意の付加的／異なるコードを書く必要なく、提供されてもよい。

【００３５】

ある実施形態は、企業サーバ１２０自体に、ライブラリ（例えば、動的リンクライブラリ（ＤＬＬ））の形態で低レベル「抽象インターフェース」を公開させ、消費させることによって、一貫したパッケージＡＰＩ５３２を有効にし得る。その抽象クラスの二値画像が、企業サーバ１２０上のサーバ実行ファイル自体と同一のディレクトリの中に位置してもよいとともに、そのインターフェースライブラリの同一バージョンが、パッケージインターフェースソフトウェア開発キットを介してパッケージ開発者に公開される。それぞれのプロジェクトをパッケージインターフェースライブラリに直接リンクし、その中で抽象クラスをインスタンス化することによって、企業サーバ１２０は、展開のために内部ディレクトリの中に到達するときに、パッケージ開発者のパッケージ（二値画像）が何を含有するであろうかを検出することができる。

【００３６】

加えて、企業サーバ１２０は、それとリンクし、企業サーバ１２０上のライブパッケージとしてそれをロード／展開することに先立って、（例えば、ＤＬＬにクエリを行うためにＭｉｃｒｏｓｏｆｔ（ＴＭ）．ＮＥＴ内観を使用して）有効性および完全性についてライブラリにクエリを行うことによって、パッケージ５３０の開発者が全ての要求された抽象クラス方法を実装したことを検証してもよい。開発者が、標準開発者ツールを使用して、不完全なライブラリを作成または構築しようとするために苦労するであろう一方で、開発者は、潜在的に、非標準開発者ツールを用いて部分的／断片化ライブラリを構築し得る。ある実装では、有効性についてクエリを行うことは、公開または予期ライブラリクラスインターフェース実装への内部（例えば、同一企業サーバ内の）バイナリハードリンクと組み合わせられてもよい。これは、企業サーバ１２０がパッケージ５３０を公開し、同時にパッケージ５３０に無数のサービス（例えば、単一サインオン、企業メタデータディクショナリ、Ｈｅａｌｔｈ　Ｌｅｖｅｌ－７積分点、および他のサービス）を提供することを可能にするように、パッケージＡＰＩ５３２を補助してもよい。

【００３７】

ある実装では、サービスＡＰＩ５２０は、利用可能なオプションの参照を行い、それらを呼び出しアプリケーション５０に返してもよい。これは、動的ＩＤＬ機能を通して行われてもよい。参照は、どのようなパッケージおよびサービス５２５がインストールされるか、またはアプリケーション５０に利用可能であるかに関して、リアルタイム更新を与えてもよい。例えば、アプリケーション５０は、利用可能なパッケージおよびサービス５２５を要求してもよく、サービスＡＰＩ５２０は、どのようなオプションが利用可能であるかをユーザまたは呼び出しアプリケーション５０が把握したい、プログラミングレベルを規定するように、ドット表記に基づくオプションのリストを返してもよい。実施例として、ユーザは、そのバージョン１．０、１．１、および２．０がサーバ上で利用可能である、ソフトウェアの１．１バージョンを実行していてもよい。ユーザは、１．１バージョンに関する利用可能な動作を要求し、ソフトウェアの１．１バージョンに利用可能な公開エンドポイントのリストを受信してもよい。

【００３８】

本アプローチを実装する代替的方法は、アプリケーション５０が、任意の所与の企業サーバ１２０上で利用可能なサービスの動的リストをインデックス化または調査し、次いで、これらのサービスをその場限りで消費することを可能にすることであろう。本タイプの機能性は、例えば、利用可能なバイナリ拡張のディレクトリリスト（例えば、ＰＨＰ；ハイパーテキストプリプロセッサ（ＰＨＰ）実行ファイル）を通して、またはＸＭＬおよびＵｎｉｖｅｒｓａｌ　Ｄｅｓｃｒｉｐｔｉｏｎ，　Ｄｉｓｃｏｖｅｒｙ　ａｎｄ　Ｉｎｔｅｇｒａｔｉｏｎ（ＵＤＤＩ）を使用して、達成可能である。これらのサービスの展開は、一次元的であり、元の展開の企業サーバ１２０およびプロトコル上のみで利用可能であり得る。

【００３９】
加えて、サービス展開エンジン５２７のある実装は、現在進行中の他のサービスまたは動作の配信に影響を及ぼすことなく、規則的な様式でそれらの動作（例えば、ＳＯＡＰ起動ビジネス論理）をクライアントにエクスポーズするために、全てのパッケージ５３０を動的に隔離および統合してもよい。本隔離は、パッケージがクラッシュする場合に損傷を防止または制限するために、リソースの所定の制御されたセット（例えば、メモリ、プロセッササイクル等）をパッケージ５３０に提供することによって、達成されてもよい。加えて、本隔離は、企業サーバ１２０上で起こる他の動作によって実質的に干渉される、または実質的に影響を受けることなく、パッケージ５３０が実行されることを可能にする。ある実装では、サービス展開エンジン５２７は、いかなるパッケージもサーバをクラッシュする能力を有していないことを確実にするように、ロバストなプロセス内例外および／または誤差範囲で個々のパッケージを包んでもよい。加えて、パッケージは、全ての同時に実行しているプロセスが、それらの機能を完了するための適切なリソースを有することを可能にするように、リソースを必要とするプロセスが抑制されることを確実にするために、企業サーバ１２０によって監視されてもよい。
【００４０】
図６は、ある実装による、パッケージ５３０を展開解除または再展開するための方法６０００のフローチャートを図示する。最初に、ステップ６１００では、サービス展開エンジン５２７が、パッケージ５３０に特定の動作を行う命令を受信してもよい。本命令は、種々のソースから受信されてもよい。命令は、公開エンドポイント５２６のうちの１つを経由して伝送され、例えば、図５の方法に従って処理されてもよい。ある実施形態では、命令は、例えば、企業サーバにアタッチされた端末またはワークステーションを経由して、企業サーバ１２０と直接相互作用するユーザから受信されてもよい。命令は、展開解除または上書きコマンドを含み得る、具体的パッケージ５３０上で行われる特定の方法を記述してもよい。
【００４１】
ステップ６２００では、サービス展開エンジン５２７は、パッケージ５３０が全ての未処理動作を完了したかどうかを検出するように、具体的パッケージ５３０を監視してもよい。現在動作しているパッケージ５３０に作用すること（例えば、パッケージ５３０を除去すること）は、クライアントアプリケーション５０の不安定性またはデータの損失を引き起こし得る。本検出は、例えば、パッケージ５３０のリソース使用を監視すること、またはアクティブプロセスリストを監視することによって行われてもよい。
【００４２】
ステップ６３００では、パッケージ５３０が、新しい動作を容認することを妨げられてもよい。本ステップは、パッケージ５３０がいかなるリソースを使用することも防止するサブステップ、パッケージ５３０を付加的要求に見えなくするサブステップ、パッケージ５３０の許可レベルを変更するサブステップ、およびパッケージ５３０をロックするサブステップを含むが、それらに限定されない、あるサブステップを含んでもよい。
【００４３】
ステップ６４００では、パッケージが除去されてもよい。本ステップ６４００は、パッケージ５３０を完全に除去するステップを含んでもよい。しかしながら、ある実装では、パッケージ５３０は、コンピュータ可読媒体１２３上に記憶されたままであってもよいが、例えば、ステップ６３００のあるサブステップのうちの１つを行うことの結果として、実質的に不安定な状態にとどまってもよい。
【００４４】
ステップ６５００では、受信される命令に応じて、新しいパッケージ５３０が追加または展開される必要があり得る。これは、パッケージの有効性を検証するサブステップ、パッケージをエンドポイント５２６に公開するサブステップ、パッケージ５３０をインストールするサブステップ、および他のそのようなサブステップを含んでもよい。
【００４５】

説明される方法は、パッケージおよびサービス５２５のバージョニングに配慮する動的展開を行うために利用されてもよい。例えば、本方法が特定のパッケージのバージョン３を再展開するために使用される場合には、前のバージョン３が、その呼び出しを終了し、特定のパッケージの新しいバージョン３と円滑に置換される。

【００４６】

先述の内容から、本発明の具体的実施形態が例証の目的で本明細書に説明されているが、本発明の精神および範囲から逸脱することなく、種々の修正が行われ得ることが理解されるであろう。したがって、本発明は、添付の請求項による場合を除いて制限されない。

【図１】

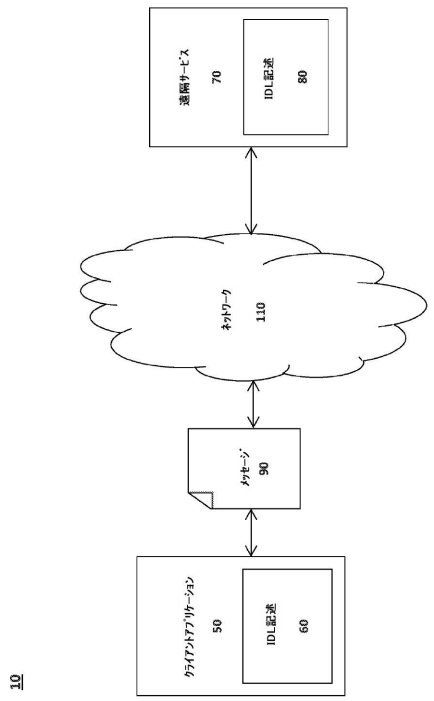

FIG. 1

【図２】

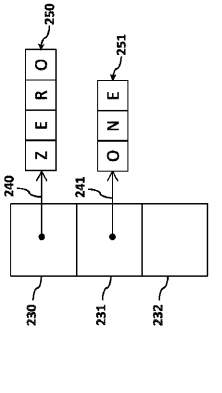

FIG. 2

【図３】

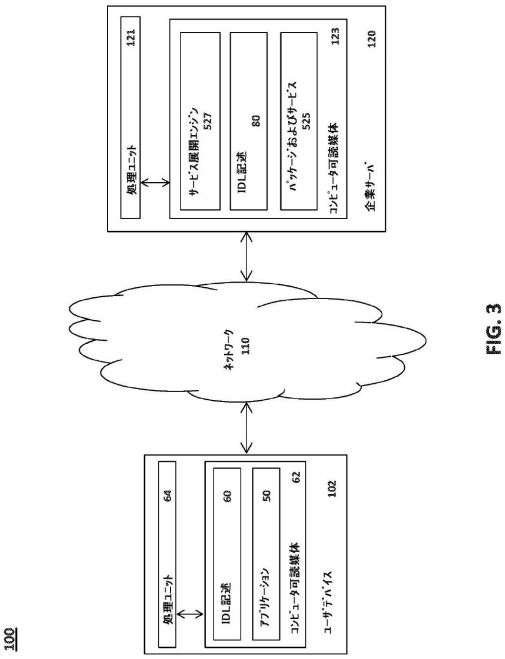

FIG. 3

【図４】

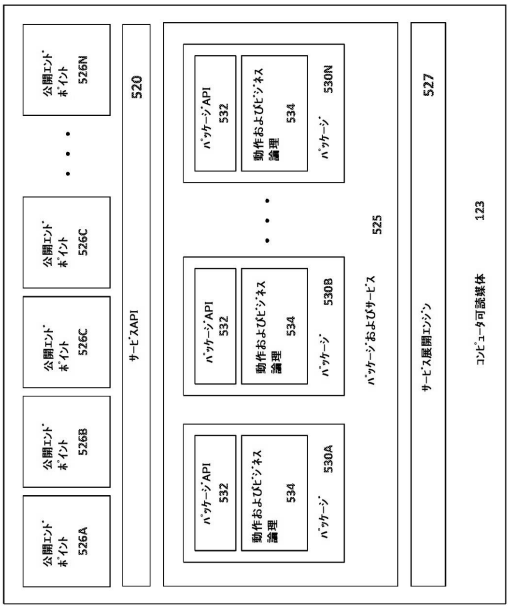

FIG. 4

【図５】

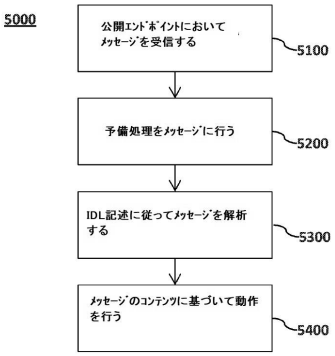

FIG. 5

【図６】

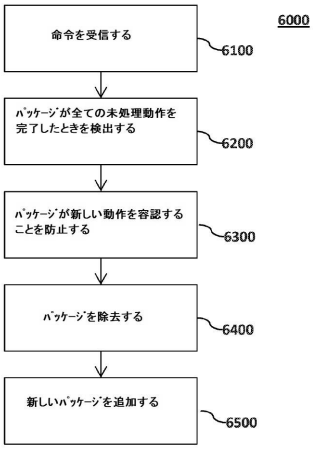

FIG. 6

フロントページの続き

(74)代理人　230113332
弁護士　山本　健策
(72)発明者　カティーブ，　ラルフ
アメリカ合衆国　ミネソタ　５５４１８－３０５４，　ミネアポリス，　ヘイズ　ストリート　エヌイー　２８５４

審査官　清木　泰

(56)参考文献　特開２０１４－１８３５８７（ＪＰ，Ａ）
特表２０１３－５４２５２４（ＪＰ，Ａ）
特開２０１３－１４５４３６（ＪＰ，Ａ）
特開２０１１－１９８２９８（ＪＰ，Ａ）
特開２００９－２１７４１０（ＪＰ，Ａ）
特表２００５－５３２６３４（ＪＰ，Ａ）
特開２００５－３４６４０８（ＪＰ，Ａ）
特開２００５－０９２４２３（ＪＰ，Ａ）
特開２００２－１８３１０６（ＪＰ，Ａ）
国際公開第２００１／０９５１４５（WO，Ａ１）
米国特許出願公開第２０１３／０００７７７３（ＵＳ，Ａ１）
米国特許出願公開第２００７／０２５５７１８（ＵＳ，Ａ１）
米国特許出願公開第２００５／００９７５６６（ＵＳ，Ａ１）
Ian Griffiths著，鈴木幸敏，外８名訳，プログラミングC#　第7版，株式会社オライリー・ジャパン，２０１３年１１月２８日，初版第1刷，Pages:139-142
Chris Smith著，鈴木幸敏訳，プログラミングF#，株式会社オライリー・ジャパン，２０１０年８月２０日，初版第1刷，Pages:106-107

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　９／４５５－９／５４
Ｇ０６Ｆ　８／００　－８／３８
Ｇ０６Ｆ　８／６０　－８／７７
Ｇ０６Ｆ　９／４４　－９／４４５
Ｇ０６Ｆ　９／４５１
Ｇ０６Ｑ１０／００　－１０／１０
Ｇ０６Ｑ３０／００　－３０／０８
Ｇ０６Ｑ５０／００　－５０／２０
Ｇ０６Ｑ５０／２６　－９９／００
Ｇ１６Ｚ９９／００